MOODLEWATCHER: ONE YEAR EXPERIENCE OF DETECTING AND PREVENTING FRAUD WHEN USING MOODLE QUIZZES

Rodolfo Matos, Filipe Carvalho, Sofia Torrão, Tito Vieira

University of Porto, Faculty of Engineering (FEUP),Computer Centre (CICA) (PORTUGAL)
rmatos@fe.up.pt, filipec@fe.up.pt, storrao@fe.up.pt, tito@fe.up.pt

Abstract

The possibility of obtaining results that have been fraudulently interfered with always exists, in all processes of student testing and evaluation. In traditional methods of evaluation, such as exams carried out using pen and paper, any fraud that occurs is extremely difficult to detect after it has taken place. The use of e-learning environments such as Moodle, however, may bring benefits that can be applied to the evaluation process in response to this problem. When using Moodle, all actions carried out by participants (regardless of whether they are teachers or students) can be automatically logged by the system, and when carrying out online quizzes this detailed information can be used as an aid in the detection of non-authorized situations that occurred during the process. The aim of this article is to present a tool, and the results of its usage over one year period, developed by the FEUP e-learning team, that permits the auditing and visualization of these situations in quizzes carried out on the Moodle@FEUP platform, thereby permitting the identification of potential offenders in this area. We also present an approach that helps to mitigate some kind of Denial-of-Service attacks to Moodle infrastructures based on Nginx**,** Apache and MySQL. Taking on board the most recent strategies and contexts of testing and evaluation, structural alterations to Moodle are also proposed which will enable the implementation of mechanisms to prevent these new methods of fraud.

1 INTRODUCTION

At the Faculty of Engineering of the University of Porto (FEUP), mechanisms to prevent and improve security in computer-based exams have been in use for quite some time now. Some of these work as a complement to Moodle, and are regularly requested and used by the teaching staff. From amongst these mechanisms we can highlight Network Access Restriction (Restrição de Acesso à Rede - RAR), which completely blocks all access, whether physical or virtual, to the intranet or Internet. By using the RAR tool with Moodle, teaching staff can restrict an exam to a specific location (e.g. to room B104), with a specified time frame and also choose which network services they want active. An example of this type of evaluation context would be an exam that has its written component set on Moodle, and which must be answered on Moodle but that permits the use of a determined application for calculations - e.g. the calculator. In this scenario, the teacher needs to create the quiz activity on Moodle, configuring it as if it were an exam (one try only, secure window, and other Moodle configurations), reserving the room for the exam and requesting the security conditions required - Moodle exam with access being allowed only to Moodle and to the calculator. On the date that the exam is to be held, the room(s) is set up for the exam by the CICA team, and the exam takes place in a controlled environment in which all the computers in the allocated rooms only have access to the Moodle server and local access to the authorized applications. Despite these mechanisms, certain situations have recently been detected which demonstrate the creativity shown by students in relation to bending the rules, a spirit which has not been defeated by the introduction of new technologies for evaluation and testing. Access to Moodle, which is permitted for the taking of the exam, also permits access to other Course Units (the Unidade Curricular – UC) with areas in Moodle as well as to other activities and resources within the Course Unit to which the exam itself relates. This gives rise to situations in which documents and resources are accessed (such as discussion forums or previous evaluations...) that might be considered to be "unauthorized" elements of consultation, but that students are able to access. As well as the accessing of resources for consultation, students have also been detected exchanging information as well as taking or looking at more than one exam. The Moodle platform does not yet offer a full solution for that problem.

To respond to these situations, and to deal with the challenges that we currently face, it is of utmost importance that we analyse the information conveyed by Moodle logs, an activity that may contribute

decisively to the detection of unauthorized or even fraudulent situations. MoodleWatcher is an auditing tool for Moodle logs that facilitates the visualization of these situations.

The document that we hereby present is divided into three parts. The first part focuses on giving a more detailed description of the situations of possible fraud that have already been detected and diagnosed, and the second part in which we present the MoodleWatcher tool, with examples of how it may be used to detect this type of circumstances. The third part focuses on describing some situations that occurred during the last year of usage of the system, including some Denial-of-Service attacks and ways of mitigating them. The document ends by considering the work that still needs to be done, and draws conclusions on work done so far.

2 EXAMPLES OF FRAUD IN MOODLE TESTS

Traditional forms of cheating in exams frequently include "whispering the answer to your neighbour", "crib sheets" and "looking at somebody else's work". We find, on Moodle, methods of cheating that are identical in methodology, but to which we attribute different names:

2.1 Exchange of exams between students

This form of cheating is, according to our records, currently the most popular. The basic method essentially consists of the sharing of the Moodle access code amongst the various participants in the fraud, which allows them to "tell each other the answers" for the exam.

2.2 Use of different accounts from the same workstation

This method consists of accessing one or more different accounts that do not belong to the actual student. This form of fraud is the method used to "take the exam for a friend", and may also be used to see what another student has done.

2.3 Most popular methods of (unauthorized) consultation are:

2.3.1 Looking at blogs

The consultation of blogs is the second most popular method. Students, when accessing their profile, are able to access a method of sharing information that is identical to the archaic system of "pen - paper", but this time using the more modern method of "copy - paste".

2.3.2 Looking at forums

The third most popular method, as with the consultation of blogs, permits the exchange of information between students as well as the simple access of information that has been placed in the system on a previous occasion.

2.3.3 Looking at other resources and suspicious behaviour

If even the sharing of information via a Wiki - which amazingly includes the history of all alterations made to the page! – is used by some students to cheat, what can be said for all the other resources that are available on the platform? Needless to say, anyone who spends 30 minutes of a test refreshing the page of a blog belonging to another student is not exactly thinking of taking the test by themselves.

3 THE MOODLEWATCHER TOOL

MoodleWatcher consists of a front-end web application that shares code with Moodle in relation to the use of database access. This integration is, however, limited to the absolute minimum that would enable the tool to be used by the majority of Moodle versions, regardless of their age.

The main purpose of the tool is to provide a simple and efficient monitoring method to be used by teachers in order to assure them of the integrity of the tests and exams carried out using Moodle.

Not all actions carried out on Moodle have a local existence (such as a paper that can be destroyed). The activities of all Moodle users are registered for posterity in a series of actions that can be identified in relation to place and time. The problem up to now was to be found in the actual analysis of

these logs, given their size and complexity. MoodleWatcher brings something new to this situation, as it is a method of collating logs in such a way as to permit clear and unequivocal identification - when conformity tests have been carried out - of how and by whom a case of fraud was committed.

When the use of MoodleWatcher is required, a set of conformity tests must be carried out, without which the reliability of the system cannot be guaranteed:

o A test on Moodle does not permit more than one attempt to be made;

o The time of starting and finishing are already defined;

o IP restriction (subnet) is set by the RAR system.

Besides these four fundamental rules, the same test must obviously not be used for different 'shifts' of an exam, in which different students sit down at the same computer, and where the start and finish times that have been set actually relate to the start of the first 'shift' and the end of the last one. It is also assumed that the computer that students are using has not been used previously by other students without having been cleaned of any type of file that can be shared (such as by making a print screen of the answers, or saving the answers on a text file to be used by the person who next sits at that computer).

The system can be used in real-time or as a tool for auditing after the exam has taken place. In either case, the teachers responsible for the Course Unit will be in possession of an attendance sheet, which provides a complete list containing the photograph, name and ID of the students, which can be printed out and then given to the students to sign. This is the method used to validate the presence of the student in the room and on the computer the IP of which has been attributed to them.

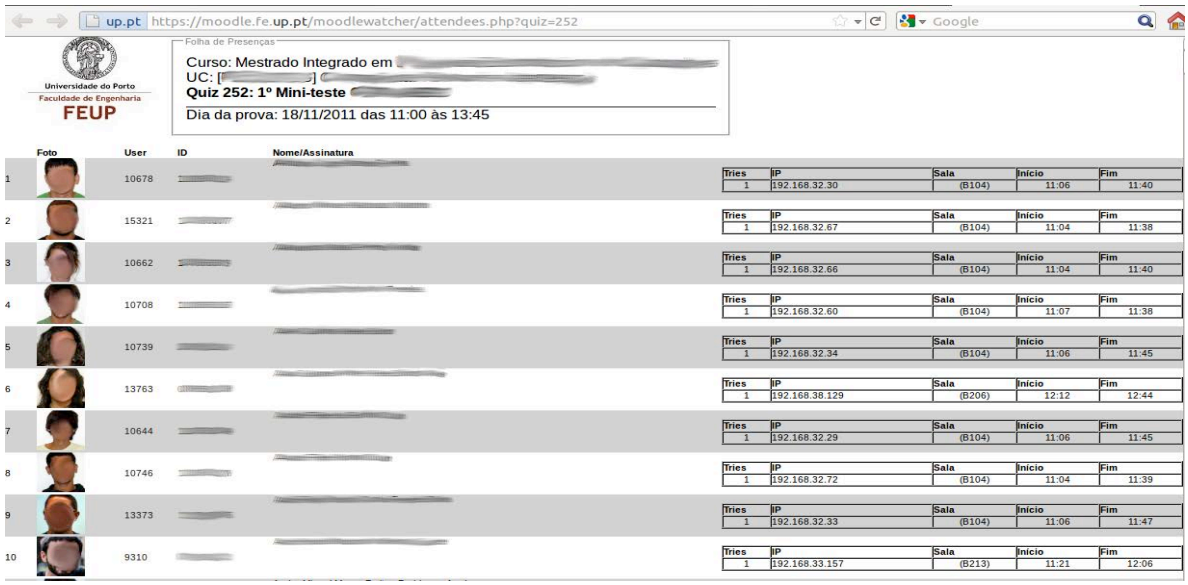

Figure 1: Attendance sheet

Whenever the MoodleWatcher page that relates to the quiz in question is refreshed, the system immediately recalculates all the information available and shows an "ATTENTION!" or "Warning!" message in relation to the cases that have been identified as being suspect, classified by degree of severity.

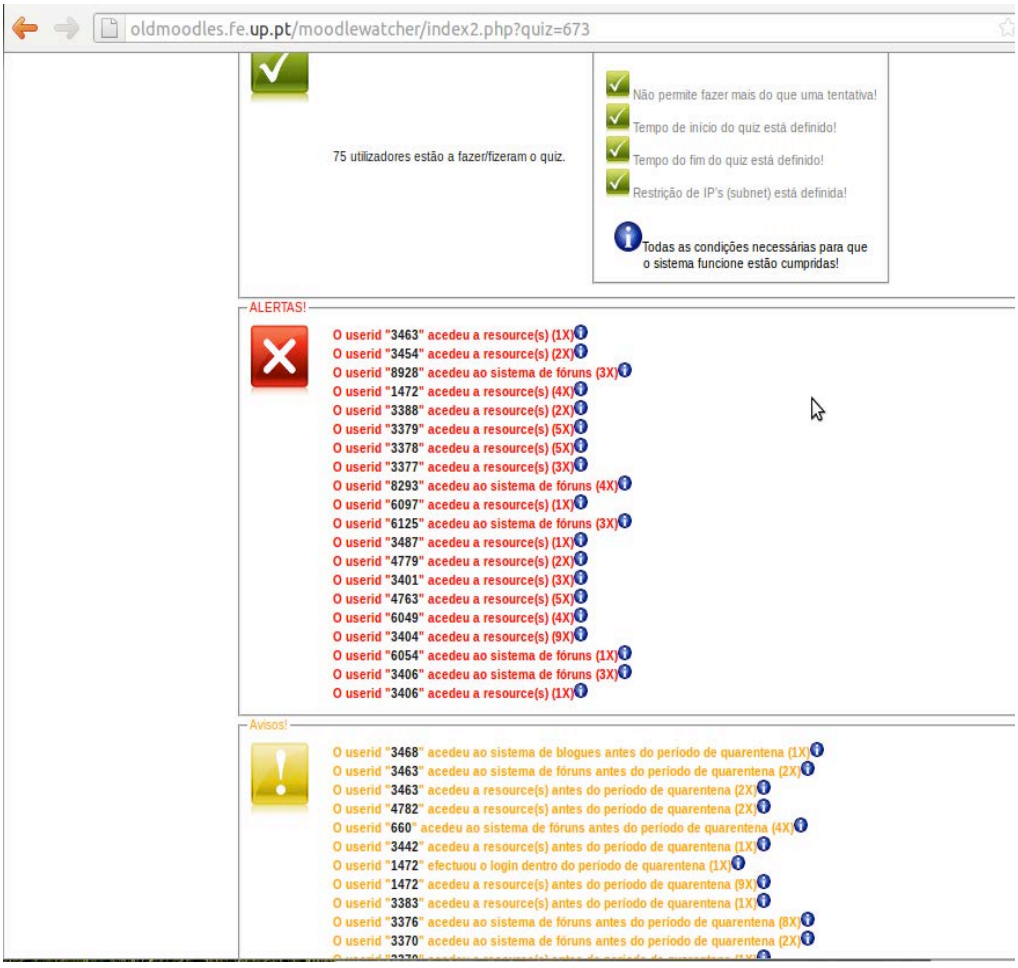

Figure 2: Dashboard view.

With "Warning!" messages, less severe cases would include situations such as:

- 'User ID X has logged in BEFORE the start of the quiz!' which indicates that the student has accessed the room in which the test is to be held before the security perimeter has been set up, allowing them to access (and save on the computer) all kinds of information.

- Other types of warning include "access to the forum/blog/resource/etc. systems BEFORE the quarantine period". The quarantine period is the period of time set out within the definitions of the test, in which a student must not under any circumstances access any type of information except for the test itself. If, immediately before starting the test, a student accesses this type of information, this means that they will potentially have this information available to them during the time that they are taking the test.

All anomalous situations that are detected during the quarantine period will result in the issue of an "ATTENTION!" message. A typical example would be an entry of the type:

- 'User ID "X" has accessed the forum system (3X)', in which the "3X" indicates the number of times that the situation has been detected.

MoodleWatcher may also be used as a tool for the monitoring and auditing of Moodle tests in order to ensure their integrity, allowing a teacher to analyse the path followed by the student. This path is presented in order of time, so as to better highlight any doubts or false positives. At this stage it is best to point out that there may be valid reasons for the above-mentioned situations, and that it is up to the person in charge of the course to distinguish between real episodes of fraud and cases of 'mistaken identity' by looking at the situations in context.

4 ONE YEAR USING MOODLEWATCHER

4.1 Infrastructure background

Our normal usage of a Moodle platform evolves around 10.000 active users, enrolled in more than 1.100 courses. Some of the courses deployed have around 500 student enrolments.

The FEUP Moodle system previously had a balancing system based on a physical Alteon load balancer. This Alteon system provided two main features: load balancing and SSL certificate hosting. The system was supported by three web back-ends, powered by Apache.

SSL servers are currently the only way to securely process confidential transactions and secure user authentication. But is often the case that Moodle instances deployed have not HTTP communications encrypted. Thus it makes extremely easy to hijack user's credentials.

Load balancing enables us to distribute workload across multiple computers, thus enabling us to increase reliability through redundancy, and minimize response time.

With the usage by the community increasing rapidly and the understandable system load that came with that usage, the alteon system gradually started showing signs of not coping, especially in periods of extremely high load, such as when exams were taking place. Other problems were also identified, such as the link of the machines was restricted to 100 Mbps. With the FEUP's Network being upgraded to Gigabit this created a bottleneck on the system. Another problem was the fact that the back-end machines were directly accessible by all of the network, which in terms of security, was not desirable.

By the academic year of 2010/2011, the system was presenting an unacceptable number of downtime due to high load. So a new implementation was devised.

4.2 Logistic and other problems

When the number of students that are scheduled to attend Moodle quizzes is too numerous to fit in a single room – which is often what happens in our campus - we must have some logistic tricks under the sleeve to tackle the problem. Those include:

Split the students across different rooms

Use a shift system

Use a mix between (1) and (2)

The recurrent problem that we had with (1) was due to the fact that for securing the network addresses, we needed to pass that information to the teachers in a sensible and easy way, that simply was not available. In our facilities is not uncommon to use five (or more) different rooms for the same exam. For those cases, teachers used to be given a range of IP's in a string like:

192.168.39.2, 192.168.39.8-31, 192.168.39.160-173, 192.168.38.54, 192.168.38.60-83, 192.168.33.164-168, 192.168.33.170-195, 192.168.33.50-97, 192.168.33.98-135,

This is obviously a very error prone way of proceeding, and had become one of the major headaches of our help-desk personnel. Because of that, we changed Moodle's “/lib/moodlelib.php” function “address_in_subnet”. With that, we were able to use some network macros that could easily identify the room used. The change can be done by inserting a conditional statement in the following way:

```
function address_in_subnet($addr, $subnetstr) {
// (START) - EXAM ROOMS SUBNETS
  $subnets = explode(',', $subnetstr);
   foreach ($subnets as $subnet) {
      $subnet = trim($subnet);
      if ($subnet === '') {
         continue;
      }
      switch ($subnet) {
        case 'B104':
          $subnetstr = $subnetstr . ', 192.168.32.1-47, 192.168.32.49-74';
          break;

        case 'C42':
          $subnetstr = $subnetstr . ', 192.168.38.8-31, 192.168.38.150-165';
          break;
        case 'LAB101':
```

That small hack, enable users to use the syntax “B301, C42, LAB101” instead of the corresponding huge IP's addresses string, thus making it much more manageable and easy to debug in case of problems. After using this very successfully for more than a year for our 29 exam rooms, we made a more versatile version, so it could be used as part of Moodle's 2.x Administration configurable options.

The problem with (2) was simply to be able to clean up the exam room computers, so students would not be able to hand over to other students, documents or informations in the same workstation. Was also because of that, teachers were informed that in case of using a shift system, they should use different quizzes for each turn. That way the correct identification of the students present on each turn would be also assured.

5 TYPES OF ATTACKS WE HAD TO DEAL WITH

Injection of malicious code by Cross-Site-Scripting attacks (XSS) is something that any e-learning infrastructure is always on risk of being affected, especially if the infrastructure is used to evaluation purposes.

When one intend to use Moodle to evaluation purposes, we need to take into account that the entire infrastructure is a huge target to attacks that:

could prevent that a exam would take place (e.g. Distributed Denial-of-Service (DDoS) attacks)

allow students to circumvent the established security perimeter, in such a way that they could access forbidden contents (e.g. Cross-Site-Scripting (XSS) attacks)

These problems affect not only Moodle, but all kind of dynamic sites, Content Management Systems (CMS), Learning Management Systems (LMS) et al.

Looking at the documentation of the CMS Joomla (http://www.joomla.org/), we find detailed documentation in how to configure a Apache server in such a way that will avoid a lot of the known attacks (http://docs.joomla.org/Htaccess_examples_(security)). Nevertheless, the corresponding documentation offered to Moodle administrators (e.g. http://docs.moodle.org/22/en/Category:Security) does not cover these problems.

We should point out, that in a “out-of-the-box” Moodle infrastructure (e.g. Moodle deployed in a LAMP environment without mod_security running) and following docs.moodle.org orientations, or in a even simpler way, using “apt-get install moodle” in a Debian or Ubuntu system), is very simple to create a DoS attack, using any browser that have access to it. It is just a matter of putting the URL <moodle_URL_address>/admin/index.php in a browser, and just after pressing ENTER, keep the reload button (usually F5 key) pressed for 5~10 seconds. Usually that simple procedure would generate a huge number of requests to the database, but since the page breaks the connections, the requests would remain “on hold”. Since the default configuration of database management systems like MySQL that would serve those requests are defined to keep those requests for a few hours, those few seconds are more than enough to exhaust the number of legitimate connections slots available, thus creating a Denial-of-Service.

6 INFRASTRUCTURE

The new infrastructure is completely hosted on FEUP's Cloud Service, so it can easily be molded to increase and/or decrease its composition according to specific needs.

```
upstream moodle {
   ip_hash;
   server 172.16.20.184:80;
   server 172.16.20.185:80;
   server 172.16.20.186:80;
   server 172.16.20.187:80;
   server 172.16.20.188:80;
`
```

Load balancing is managed by a Linux virtual machine with only 2 GB of RAM. Nginx is used for the purpose of balancing both port 80 (HTTP) and 443 (HTTPS), with SSL certificate hosting. This is done by using Nginx's HttpUpstreamModule, which provides a simple round-robin load balancing. Using the ip_hash directive we can make sure the client gets forwarded to the same server every-time, so not to cause session problems.

For extra security, mainly to prevent DDoS attacks, we also use Nginx's HttpLimitReqModule, which allows us to limit the number of requests per second from a specific IP address.

```
http {
   limit_req_zone  $binary_remote_addr  zone=one:10m   rate=20r/s;
}
```

```
-A POSTROUTING -s 172.16.20.0/24 -j MASQUERADE
-A POSTROUTING -s 172.16.20.188/32 -o eth0 -j SNAT --to-source 193.136.28.161
-A POSTROUTING -s 172.16.20.187/32 -o eth0 -j SNAT --to-source 193.136.28.161
-A POSTROUTING -s 172.16.20.185/32 -o eth0 -j SNAT --to-source 193.136.28.161
```

The back-end servers are the same as in the previous infrastructure, running Linux and Apache. They are now in an isolated network. They can only communicate with the rest of the network using the load balancing server as a gateway. This can be configure using Linux's IPtables NAT features.

7 CONCLUSIONS

We compared the occurrences during quizzes done over several courses that had the same kind of conditions in these last two academic years. To acquire the data we used the output of the script do_i_need_moodlewatcher.php.

Earlier this academic year, all students were told that the system MoodleWatcher would be monitoring their evaluation processes, and they could have disciplinary consequences if they were caught cheating. However, the data we have available clearly shows, that in all situations with the exception of course “C”, the number of occurrences increased instead. These occurrences should be seen very carefully, since they were not committed by a larger group of students. The fact is that the amount of students that committed unauthorized actions, have done them in a larger scale.

	2011-12		2010-11		
Course	Students	Occurrences	Students	Occurrences	Variation on ocurrences
A	18,8	5,9	11,0	3,0	4,06%
B	214,0	325,0	226,0	110,0	103,20%
C	**242,0**	**2,0**	**301,0**	**12,0**	**-3,16%**
D	137,0	36,0	130,0	4,0	23,20%
E	72,0	67,7	112,3	2,0	92,20%
F	112,0	102,0	97,0	1,0	90,04%
G	83,2	7,8	86,5	2,5	6,48%

The difference with course “C” evaluation was because the entire evaluation process began shortly after some of the students had suffered disciplinary actions upon them.

This tendency clearly shows that only announcing the measures is not enough. If the potential offenders feel that the “cost-benefit” of obtaining results that have been fraudulently interfered with is worth it, they will continue to do so, despite all warnings. To avoid these situations, hardening Moodle, closing the open wide doors identified is – obviously - a top priority. Nonetheless, one should ensure that disciplinary actions are not to be taken lightly by the students. If that is not the case, there will be always potential offenders that will try to test the effectiveness of the system.

REFERENCES

[1] Matos, R., 2011, Do I need MoodleWatcher Audit Report? test script, *How to catch/avoid quiz cheating students Community Discussion forum*, http://moodle.org/pluginfile.php/134/mod_forum/attachment/817464/do_i_need_moodlewatcher.php (October, 2011)

[2] Matos, R., Torrão, S., Vieira, T., 2012, Proceedings of INTED2012 Conference, pp.4997-5001, ISBN: 978-84-615-5563-5